**College of Engineering Pune**

**DBMS Project**

**Pharmacy Management System**

**Synopsis**

**Vishal Sharma 111903159**

**Tejal Khairnar 111903157**

**Shubham Boke 11803182**

* 1. ***Problem Statement***

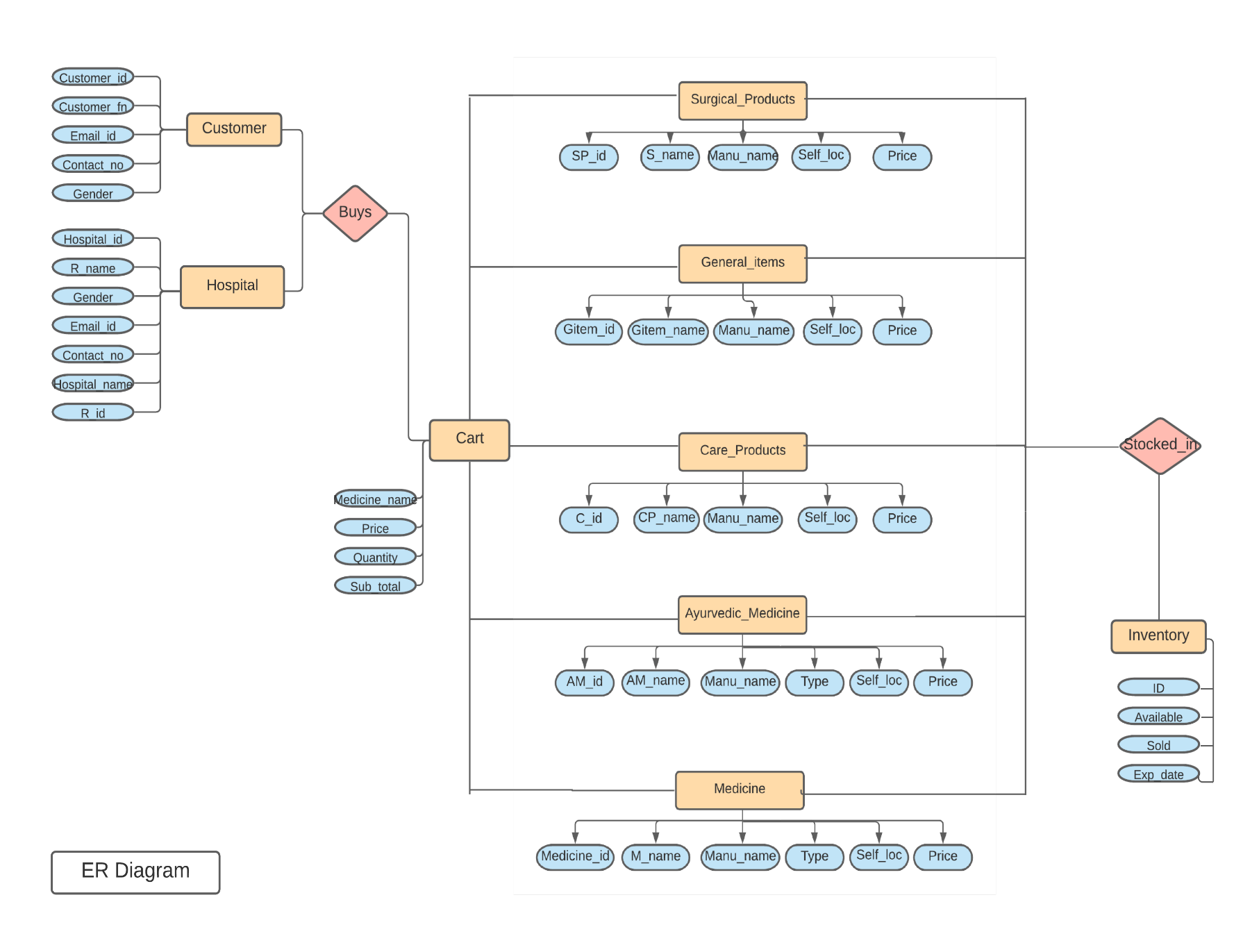
The world changes every-day and there are new medicines discovered. The main problem in a Pharmacy Management System is to keep it updated. Due to the constant advancement in the medical field of the world, it is very difficult to keep the pharmacy management system up-to-date and to keep a track of every medicine available. It is a very tedious job to keep the system proper and error free and to keep the pharmacy working in an efficient and in an orderly manner. Medicines cures diseases and illnesses and hence are very important to be updated every now and then. Expiration dates create an issue as well since the pharmacy needs to be careful while distributing medicines in large numbers.

* 1. ***Functional requirements of the system***

The project, Pharmacy Management System, covers various functional requirements. Some of them being- log-in, sign-up, profile, etc.

|  |  |  |
| --- | --- | --- |
| S.No | FUNCTION | DESCRIPTION |
| 1. | Log-in | The log-in function is used to provide absolute privacy to a customer or a specific hospital by asking its ID- number and name. The whole system is personalized according to the customer/hospital that has logged in. |
| 2. | Sign-up | The sign-up function helps a customer/hospital to create a new account in the system and then move further with the shopping process. |
| 3. | Profile | This function shows all the details of the logged in entry. Details like name, contact number, email-ID, ID, address, etc. are displayed. |
| 4. | Edit Profile | Edits the details if the users need to alter their details. All details except ID of the user can be deleted. |
| 5. | Product Buttons | There are products that are segregated into 8 types and their buttons when clicked will display the items for that category that can be added in the cart. All items are displayed on the main page with a paging system with one page showing 4 items maximum. |
| 6. | Cart | The cart function is like any other cart where the customer’s final shopping details are visible. This allows the customer/hospital to review its order and place it and further move towards the payment. |
| 7. | Log-out/Exit | This function is used to finally exit the whole system. Once exited, the customer/hospital is automatically logged out as well. |

***2.1 ER Model***

******

***2.2 Reduction of ER model to Relational Model***

* temp\_cart (M\_name, Price, Quantity, Sub\_total)
* ayurvedic\_med (AM\_id, AM\_name, Man\_name, Type, Self\_loc, Price)
* care\_products (C\_id, CP\_name, Manu\_name, Self\_loc, Price)
* customer (Customer\_id, Customer\_fn, Customer\_ln, Email\_id, Contact\_no, Gender)
* general\_items (GItem\_id, GItem\_name, Manu\_name, Self\_loc, Price)
* medicine (Medicine\_id, Pres\_no, M\_name, Manu\_name, Type, Self\_loc, Price, Prescription)
* hospital (Hospital\_id, Hospital\_name, R\_name, R\_id, Email\_id, Contact\_no, Gender)
* inventory (ID, Available, Sold, Exp\_date)
* surgical\_products (SP\_id, S\_name, Man\_name, Self\_loca, Price)

Functional Dependencies: -

1. **Customer**

(Customer\_id)+ = {Customer\_fn, Email\_id, Contact\_no, Gender}

1. **Hospital**

(Hospital\_id, R\_id)+ = {Hospital\_name, R\_name, Gender, Email\_id, Contact\_no,}

1. **Medicine**

(Medicine\_id)+ = {M\_name, Manu\_name, Type, Self\_loc, Price}

1. **Ayurvedic\_Medicine**

(AM\_id)+ = {AM\_name, Manu\_name, Type, Self\_loc, Price}

1. **Care\_Products**

(C\_id)+ = {c\_name, Manu\_name, Self\_loc, Price}

1. **General\_items**

(Gitem\_id)+ = {Gitem\_name, Manu\_name, Self\_loc, Price}

1. **Surgical\_Products**

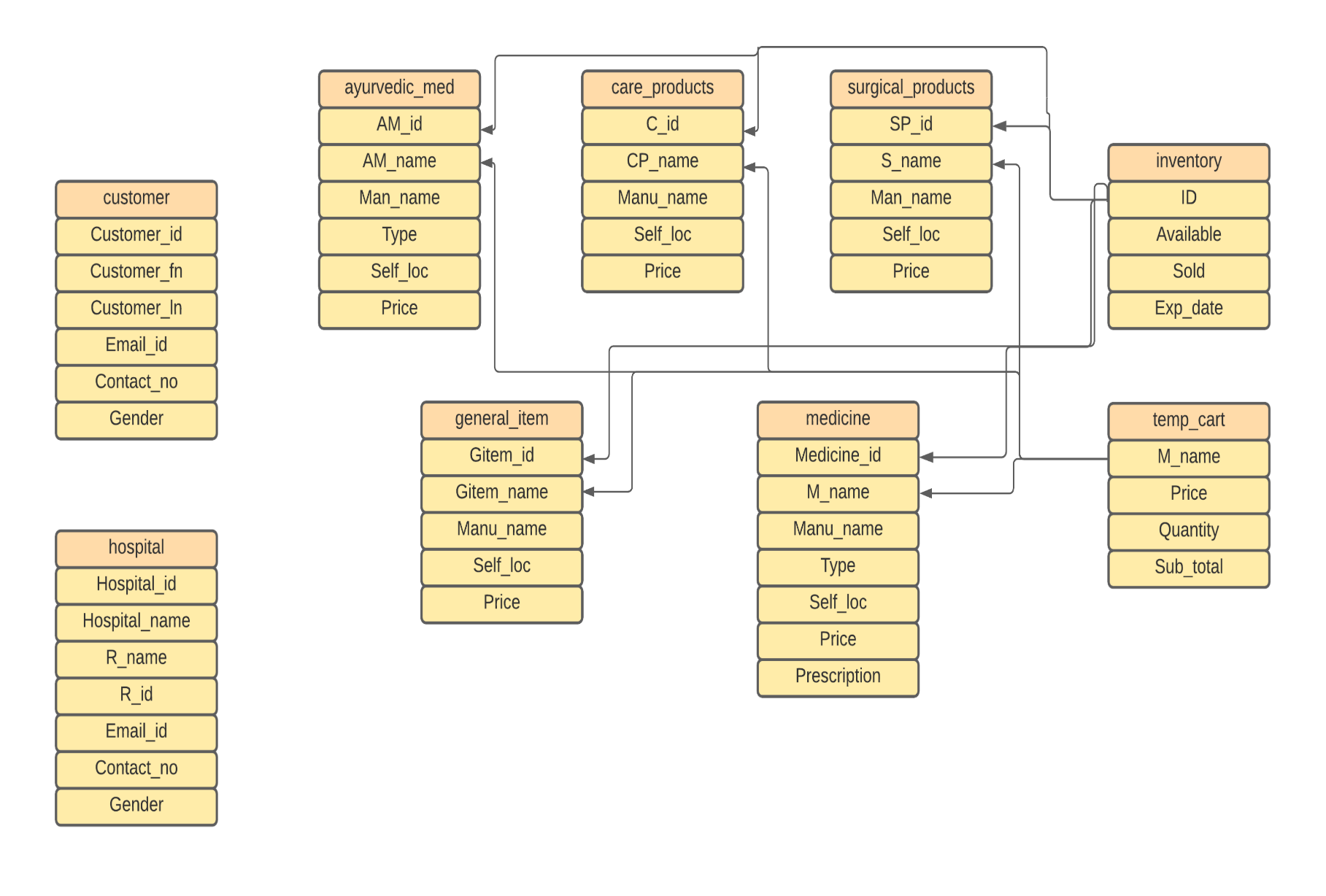
(SP\_id)+ = {s\_name, Manu\_name, Self\_loc, Price}

1. **Cart**

(Medicine\_name)+ = {Price, Quantity, Sub\_total}

1. **Ayurvedic\_Medicine**

(ID)+ = {Available, Sold, Exp\_date}

*** 2.3 Schema Diagram***

**Normalization techniques applied on relational model:**

* A Relation Will be 1NF if it contains an atomic value. It should not contain repeating Groups.

**In our case 1NF is not satisfied as there are excess null valued data.**

* temp\_cart (M\_name, Price, Quantity, Sub\_total)
* ayurvedic\_med (AM\_id, AM\_name, Man\_name, Type, Self\_loc, Price)
* care\_products (C\_id, CP\_name, Manu\_name, Self\_loc, Price)
* customer (Customer\_id, Customer\_fn, Customer\_ln, Email\_id, Contact\_no, Gender)
* general\_items (GItem\_id, GItem\_name, Manu\_name, Self\_loc, Price)
* medicine (Medicine\_id, Pres\_no, M\_name, Manu\_name, Type, Self\_loc, Price, Prescription)
* hospital (Hospital\_id, Hospital\_name, R\_name, R\_id, Email\_id, Contact\_no, Gender)
* inventory (C\_id,Medicine\_id,PM\_id,SC\_id,HH\_id,Genral\_id,SE\_id, Available, Sold, Exp\_date)
* surgical\_products (SP\_id, S\_name, Man\_name, Self\_loca, Price)

Here in the inventory table , we have id of all categories and when id of one value in used to keep track of its stock , all other colums of id will have null value, therefore increasing useless data.

This is NF1 normalised by creating a single ID general in the inventory table for all products in the pharmacy. It reduces all the null values of the table.

* temp\_cart (M\_name, Price, Quantity, Sub\_total)
* ayurvedic\_med (AM\_id, AM\_name, Man\_name, Type, Self\_loc, Price)
* care\_products (C\_id, CP\_name, Manu\_name, Self\_loc, Price)
* customer (Customer\_id, Customer\_fn, Customer\_ln, Email\_id, Contact\_no, Gender)
* general\_items (GItem\_id, GItem\_name, Manu\_name, Self\_loc, Price)
* medicine (Medicine\_id, Pres\_no, M\_name, Manu\_name, Type, Self\_loc, Price, Prescription)
* hospital (Hospital\_id, Hospital\_name, R\_name, R\_id, Email\_id, Contact\_no, Gender)
* inventory (ID, Available, Sold, Exp\_date)
* surgical\_products (SP\_id, S\_name, Man\_name, Self\_loca, Price)
* A Relation Will be 2NF Only if it is 1NF and and it is Fully Functional Dependent on the Primary Key.

**-In our case each and every Attribute is Fully Functional Dependent on the**

**respective Primary Key. So, 2NF is Satisfied and hence, there will be no**

**Decomposition of the tables***.*

* A Relation will be in 3NF if it is in 2NF and no transitive dependency exists.

**-In our case no transition dependency exists. So, 3NF is Satisfied. Hence,there will be no Decomposition of the tables.**

* A Relation will be BCNF if it is in 3NF and For Each Functional Dependency ( X → Y ),X is a Super Key.

**-In our case BCNF is Satisfied. Hence there will be no Decomposition of the**

**tables further.**

**Therefore all tables after normalisation are :-**

* temp\_cart (M\_name, Price, Quantity, Sub\_total)
* ayurvedic\_med (AM\_id, AM\_name, Man\_name, Type, Self\_loc, Price)
* care\_products (C\_id, CP\_name, Manu\_name, Self\_loc, Price)
* customer (Customer\_id, Customer\_fn, Customer\_ln, Email\_id, Contact\_no, Gender)
* general\_items (GItem\_id, GItem\_name, Manu\_name, Self\_loc, Price)
* medicine (Medicine\_id, Pres\_no, M\_name, Manu\_name, Type, Self\_loc, Price, Prescription)
* hospital (Hospital\_id, Hospital\_name, R\_name, R\_id, Email\_id, Contact\_no, Gender)
* inventory (ID, Available, Sold, Exp\_date)
* surgical\_products (SP\_id, S\_name, Man\_name, Self\_loca, Price)

**Software Requirements**: Python 3.7+(libraries use- tkinter, mysql connector), Mysql Server

**Hardware Requirements**: 2.0 GHz processor, minimum 2048 MB RAM, 30 GB Storage